Nama : Rifan Wahana Banuarda

NPM : 51422441

Kelas : 2IA11

RANGKUMAN MATERI 1-6

Pengenalan Pemrograman Berbasis Objek

Pemrograman berorientasi objek (OOP) adalah cara pemrograman yang fokus pada pembuatan program melalui objek. Objek ini memiliki atribut, karakteristik, dan kadang-kadang kondisi tertentu. OOP berkembang dari pemrograman terstruktur pada tahun 1960, yang awalnya berasal dari bahasa C dan Pascal. Pendekatan terstruktur memungkinkan penulisan program yang kompleks menjadi lebih mudah. OOP tidak hanya memperhatikan cara mengatur struktur program, tetapi juga fokus pada objek yang dapat menyelesaikan masalah. Pemrograman berorientasi objek sangat efektif saat digunakan bersama dengan Objek-Oriented Analysis And Design Process (OOAD).

A. Perbedaan Antara Pemrograman Berbasis Objek dan Pemrograman Terstruktur

1. Pemrograman Berbasis Objek menggabungkan fungsi dan data dalam kelas atau objek-objek. Ini dicirikan oleh Encapsulation, Inheritance, dan Polymorphism. Struktur program menjadi lebih ringkas dengan pembuatan objek dan kelas, yang dapat digunakan berkali-kali untuk menghemat ruang memori.
2. Pemrograman Terstruktur memecah program menjadi fungsi-fungsi dan data. Ini melibatkan urutan proses dan pengulangan. Struktur program menjadi rumit karena harus mengatur urutan proses dan fungsi-fungsi. Penggunaan kembali kode program cenderung kurang optimal.

B. Class Pada PBO

Class adalah cetak biru atau blueprint dari objek. Class digunakan untuk membuat kerangka dasar objek. Misalnya, sebuah Class bisa diibaratkan sebagai kendaraan. Sebuah kendaraan memiliki ciri-ciri seperti merek, mesin, ban, dan ciri khas lain yang membuatnya menjadi kendaraan. Selain itu, kendaraan juga dapat melakukan tindakan seperti menghidupkan atau mematikan mesin.

Contoh :

```
// Deklarasi Class
public class Kendaraan{
    // Ini Adalah Class, semua konstruktor, variabel, method berada disini
}
```

C. Objek Dalam Pemrograman

Objek dapat dibandingkan dengan catatan dalam sistem berkas. Mereka adalah entitas yang memiliki atribut, karakteristik, dan kadang-kadang kondisi tertentu. Objek dapat mewakili sesuatu dalam kenyataan seperti seorang siswa, konsep seperti merek dagang, atau visualisasi seperti bentuk huruf (font). Dalam pengembangan perangkat lunak berbasis objek, objek menyimpan keadaannya dalam variabel dan perilakunya dalam metode atau fungsi.

D. Objek Pada PBO

Untuk membuat objek, Anda dapat menggunakan kata kunci "new" yang digunakan untuk menginisialisasi objek baru. Ada tiga langkah dalam membuat sebuah objek: mendeklarasikan variabel, membuat objek baru (instansiasi), dan memanggil konstruktor.

Contoh :

```
public class Kendaraan{

    // Konstruktor Dengan Parameter
    public Kendaraan(String nama){
        System.out.println("Nama Kendaraannya Adalah "+ nama);
    }

    public static void main(String[]args){

        // Perintah untuk membuat objek jenis
        Kendaraan jenis = new Kendaraan("Pesawat Terbang");

    }

}

// Output = Nama Kendaraannya Adalah Pesawat Terbang
```

Inheritance

Pewarisan adalah cara untuk "menggunakan kembali" kelas yang sudah ada dengan menambahkan fitur baru ke dalam kelas tersebut. Ada dua jenis pewarisan:

- Pewarisan tunggal (single inheritance) terjadi ketika sebuah kelas baru dibuat dari satu kelas induk.
- Pewarisan ganda (multiple inheritance) terjadi ketika sebuah kelas baru dibuat dari dua atau lebih kelas induk.

Kelas yang memberikan warisan disebut kelas induk atau kelas dasar, sedangkan kelas yang menerima warisan (kelas baru) disebut kelas anak atau kelas turunan. Untuk menerapkan pewarisan, gunakan kata kunci "extends". Kata kunci "super" digunakan oleh kelas anak untuk mengakses konstruktor, atribut, dan metode yang ada di kelas induknya.

```
super()

super(parameter)
```

```
super.namaAtribut
super.namaMethod(parameter)
```

Encapsulation

Enkapsulasi (encapsulation) adalah cara untuk melindungi property (atribut) / metode tertentu dari sebuah kelas agar tidak dapat diakses atau dimodifikasi secara sembarangan oleh bagian program lainnya. Cara untuk melindungi data yaitu dengan menggunakan access modifiers (hak akses). Ada 4 hak akses yang tersedia, yaitu :

No	Modifier	Pada class dan interface	Pada method dan variabel
1	Default (tidak ada modifier)	Dapat diakses oleh yang sepaket	Diwarisi oleh subkelas dipaket yang sama, dapat diakses oleh method-method yang sepaket
2	Public	Dapat diakses dimanapun	Diwarisi oleh subkelasnya, dapat diakses dimanapun
3	Protected	Tidak bisa diterapkan	Diwarisi oleh subkelasnya, dapat diakses oleh method-method yang sepaket
4	private	Tidak bisa diterapkan	Tidak dapat dikases dimanapun kecuali oleh method-method yang ada dalam kelas itu sendiri

Aksesabilitas	public	private	protected	default
Dari kelas yang sama	Ya	Ya	Ya	Ya
Dari sembarang kelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari sembarang kelas di luar paket	Ya	Tidak	Tidak	Tidak
Dari subkelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari subkelas di luar paket	Ya	Tidak	Ya	Tidak

Polymorphism

- Overloading
  Diperbolehkannya dalam sebuah class memiliki lebih dari satu nama function/method yang sama tetapi memiliki parameter/argument yang berbeda.

```
public class Overloading {
    public void Tampil(){
        System.out.println("I love Java");
    }
    public void Tampil(int i){
        System.out.println("Method dengan 1 parameter = "+i);
    }
    public void Tampil(int i, int j){
        System.out.println("Method dengan 2 parameter = "+i+" & "+j);
    }
    public void Tampil(String str){
        System.out.println(str);
    }

    public static void main(String a[]){
        Overloading objek = new Overloading();
        objek.Tampil();
        objek.Tampil(8);
        objek.Tampil(6,7);
        objek.Tampil("Hello world");
    }
}
```

- **Overriding**

  Kemampuan dari subclass untuk memodifikasi method dari superclass-nya, yaitu dengan cara menumpuk (mendefinisikan kembali) method superclass-nya. Contoh overriding method dapat dilihat pada subclass “Mobil” yang mendefinisikan kembali method keterangan() dan hapus() dari class “Kendaraan”.